Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Regrouping Particle Swarm Optimization: A New Global Optimization Algorithm with Improved Performance Consistency Across Benchmarks

George I. Evers and Mounir Ben Ghalia
The Electrical Engineering Department
The University of Texas-Pan American
Edinburg, Texas USA
george@georgeevers.org, benghalia@utpa.edu

***Abstract*—Particle swarm optimization (PSO) is known to suffer from stagnation once particles have prematurely converged to any particular region of the search space. The proposed regrouping PSO (RegPSO) avoids the stagnation problem by automatically triggering swarm regrouping when premature convergence is detected. This mechanism liberates particles from sub-optimal solutions and enables continued progress toward the true global minimum. Particles are regrouped within a range on each dimension proportional to the degree of uncertainty implied by the maximum deviation of any particle from the globally best position. This is a computationally simple yet effective addition to the computationally simple PSO algorithm. Experimental results show that the proposed RegPSO successfully reduces each popular benchmark tested to its approximate global minimum.**

***Keywords*—Particle swarm optimization, stagnation, premature convergence, automatic regrouping mechanism, maintaining swarm diversity.**

I. Introduction

While stochastic, population-based, heuristic optimization methods such as particle swarm optimization (PSO) [1, 2] are less susceptible to entrapment due to their stochastic nature and reliance directly upon function values rather than derivative information, they are nonetheless susceptible to premature convergence, which is especially the case for optimization problems of high dimensionality [3]. The more communication that occurs between the population agents, the more similar they tend to become until converging to the same region of the search space. In particle swarm, if the region converged to is a *local well* containing a *local minimum*, there may initially be hope for escape via a sort of momentum resulting from the fraction of velocity carried over from the previous iteration; over time, however, particles' momenta decrease until the swarm settles into a state of *stagnation*, from which the basic algorithm does not offer a mechanism of escape.

While allowing particles to continue in a state of premature convergence may lead to *solution refinement* or *exploitation* following the initial phase of *exploration*, it has been independently observed as well as noted in [4] that after enough time, velocities often become so small that at their expected rate of decrease, even the nearest solution is eliminated from the portion of the search space particles can practically be expected to reach in later iterations.

Van den Bergh addressed this problem with his Guaranteed Convergence PSO (GCPSO) by using a different velocity update equation for the best particle since its personal best and global best both lie at the same point, which in standard global best PSO (*gbest PSO*) inhibits its explorative abilities; GCPSO is said to guarantee convergence to a local minimum [4, 5].

There is still a problem, however, in that particles tend to converge to a sub-optimal solution before reducing the cost function to its *true global minimum*. Addressing this problem, Van den Bergh developed multi-start PSO (MPSO) which automatically triggers a restart when stagnation is detected [4]. Restarting in MPSO refers to starting the search anew with a different sequence of random numbers so that even initial positions are different, and each search is independent of those previously conducted. After a pre-specified number of restarts have completed, the best solution found over all searches is proposed as the most desirable decision vector.

Following this logic, the research question is whether there might be a more efficient mechanism by which the swarm can "restart," since restarting on the original search space might cause unnecessarily repetitious searching of regions not

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

expected to contain quality solutions. In this paper, a mechanism is proposed by which the swarm can efficiently *regroup* in a region small enough to avoid unnecessarily redundant search, yet large enough to escape entrapping wells containing local minima in order to prevent stagnation. There is one continuous search, with each regrouping making use of previous information, rather than a series of independent searches.

Kennedy and Eberhart observed that if each particle is drawn toward its *neighborhood best* or *local best* instead of directly toward the global best of the entire swarm, particles are less likely to get stuck in local optima [2]. Neighborhoods in this *local best* PSO (*lbest PSO*) overlap so that information about the global best is still transmitted throughout the swarm but more slowly so that more exploration is likely to occur before convergence, thus reducing the likelihood of premature convergence [2, 6, 7]. The PSO literature seems to have focused primarily on gbest PSO due to its relatively quick initial convergence; however, hasty decisions may be of lower quality than those made after due consideration when the function being optimized is complicated. Lbest PSO still suffers from premature convergence in some cases as demonstrated somewhat severely on the Rastrigin test function, where many algorithms tend to suffer.

Wang *et al*. applied an opposition-based learning scheme to PSO (OPSO), along with a Cauchy mutation to keep the globally best particle moving [8]. The main objective of OPSO with Cauchy mutation is to help avoid premature convergence on multi-modal functions. Using opposition-based learning, two different positions – the particle's own position and the position opposite the center of the swarm – are evaluated for each randomly selected particle. Only for particles at the center of the swarm could these positions be the same.

Once the swarm has converged prematurely, there are at least five options: (i) terminate the search and accept the best decision vector found as the proposed solution, (ii) allow the search to continue and hope that the swarm will slowly refine the quality of the proposed solution, though it is likely only an approximation of a local minimizer rather than the desired global minimizer, (iii) restart the swarm from new locations (i.e. start from scratch [9]) and search again to see if a better solution can be found as in MPSO, (iv) somehow flag regions of the space to which particles have prematurely converged as already explored and restart the algorithm so that each successive search is more likely to encounter the global minimizer, or (v) reinvigorate the swarm by introducing diversity so the search can continue more or less from the current location without having to restart and re-search low quality regions of the search space.

This study focuses on the latter option since it appears to be the most efficient approach other than somehow preventing premature convergence altogether. Also, the concern here is primarily with problems of high dimensionality, where stagnation can have a severe effect on function value so that the global minimizer is not well approximated.

The main contribution of this paper consists of a new automatic regrouping mechanism embedded into standard gbest PSO, resulting in a regrouping PSO algorithm (RegPSO) that allows the swarm to escape local wells efficiently and continue in its search rather than stagnating or restarting the search entirely.

The paper is organized as follows. The standard gbest PSO algorithm is discussed in Section 2. Section 3 illustrates the premature convergence and stagnation problem using a numerical example. RegPSO is presented in Section 4. Experimental findings are reported in Section 5. Finally, Section 6 concludes this paper.

II. THE PSO ALGORITHM

The optimization problem considered herein is to

$$\text{minimize } f(\vec{x}) \tag{1}$$

where the function $f:\mathbb{R}^n \to \mathbb{R}$ to be minimized is the objective function, or cost function, of an application problem. The vector $\vec{x} \in \mathbb{R}^n$ contains the problem's decision variables. Even though (1) is considered an unconstrained optimization problem, in practice only solutions belonging to a subset $\Omega \subset \mathbb{R}^n$ are considered feasible. The feasible search space is defined by a subset

$$\Omega = \left[x_1^{\mathrm{L}}, x_1^{\mathrm{U}}\right] \times \left[x_2^{\mathrm{L}}, x_2^{\mathrm{U}}\right] \times \cdots \times \left[x_n^{\mathrm{L}}, x_n^{\mathrm{U}}\right] \subset \mathbb{R}^n \tag{2}$$

where x_j^{L} and x_j^{U} are, respectively, the lower and upper bounds of the search space along dimension j for $j = 1, 2, \cdots, n.$

For a particle swarm of size s, potential solutions to the optimization problem are given by the position vectors of the swarm's particles $\vec{x}_i = [x_{i1}, x_{i2}, \cdots, x_{in}]$, for $i = 1, 2, \cdots, s.$ The swarm is initialized by randomizing particles' positions within the search space, Ω, using random numbers drawn from a uniform distribution. Each particle keeps a memory of its own *personal best*, $\vec{p}_i(k)$, (i.e. the best position it personally has found) for its own consideration. This personal best is updated only when the particle's new position at iteration k yields a better function value than did the previously personal best at iteration $k-1$ as shown in (3).

$$\vec{p}_i(k) = \begin{cases} \vec{x}_i(k) & \text{if } f\left(\vec{x}_i(k)\right) < f\left(\vec{p}_i(k-1)\right) \\ \vec{p}_i(k-1) & \text{if } f\left(\vec{x}_i(k)\right) \geq f\left(\vec{p}_i(k-1)\right) \end{cases} \tag{3}$$

Let $P(k) = \left\{\vec{p}_1(k), \vec{p}_2(k), \cdots, \vec{p}_s(k)\right\}$ be the set of all personal bests at current iteration k. In gbest PSO, the *global best*, $\vec{g}(k)$, (i.e. the globally best position) is iteratively selected using (4).

$$\vec{g}(k) = \arg\min_{p_i(k) \in P(k)} f\left(\vec{p}_i(k)\right) \tag{4}$$

The global best is the position that has produced the smallest cost function value of all positions occupied by the swarm through current iteration k. The global best is

"communicated" via shared computer memory to all particles for consideration. Iteratively, particle i moves from its current position to a new position along *velocity* vector, $\vec{v}_i = [v_{i1}, v_{i2}, \cdots, v_{in}]$, using position update equation

$$\vec{x}_i(k+1) = \vec{x}_i(k) + \vec{v}_i(k+1) \quad (5)$$

for $i = 1, 2, \cdots, s$. The velocity is first updated as

$$\begin{aligned} \vec{v}_i(k+1) = \omega\vec{v}_i(k) + c_1\vec{r}_1 \circ \left(\vec{p}_i(k) - \vec{x}_i(k)\right) \\ + c_2\vec{r}_2(k) \circ \left(\vec{g}(k) - \vec{x}_i(k)\right) \end{aligned} \quad (6)$$

for $i = 1, 2, \cdots, s$, where $\circ$ is the Hadamard element-wise vector product, and

$k+1$ denotes the next iteration number,

k denotes the current iteration number,

$\vec{v}_i$ denotes the velocity vector of particle i,

$\vec{x}_i$ denotes the position vector of particle i,

ω is the static inertia weight chosen in the interval $[0,1]$,

c_1 is the cognitive acceleration coefficient,

c_2 is the social acceleration coefficient,

$\vec{p}_i$ is the best position vector encountered by particle i (i.e. the *personal best* of particle i),

$\vec{g}$ is the best position vector found by the entire swarm (i.e. the swarm's *global best*),

$\vec{r}_1$, & $\vec{r}_2$ are n-dimensional column vectors whose elements are independent pseudo-random numbers selected from a uniform distribution $U(0,1)$.

The value of each particle's velocity along dimension j is randomly initialized to lie within $\left[-v_j^{\max}, v_j^{\max}\right]$ and subsequently clamped to lie within the same interval since particles should only need to step through some maximum percentage of the search space per iteration. Before this was implemented, particles were prone to roam far outside the bounds of the feasible search space [7]. In most PSO implementations, the value of $v_j^{\max}$ is selected as a percentage, λ, of the range of the search space along dimension j [3]. That is

$$v_j^{\max} = \lambda \cdot \text{range}_j(\Omega), \quad (7)$$

where $\text{range}_j(\Omega)$ represents the range or length of search space Ω along dimension j. Using (2), it is calculated as

$$\text{range}_j(\Omega) = x_j^U - x_j^L, \quad j = 1, 2, \cdots, n. \quad (8)$$

The velocity clamping percentage, λ, is usually chosen to lie within $0.1 \le \lambda \le 0.5$.

While the iterative stepping process using update equations (3)-(6) continues, particles update their personal bests as they encounter better positions than encountered previously. At any point in time, the best of all personal bests is the swarm's global best shared freely between particles. The swarm eventually converges via communication of the global best and the collective movement toward it to the one best position found, thereby proposing it as a solution. The algorithm can be allowed to run either for a number of iterations expected to produce a good solution or until a user-specified criterion or threshold is reached. The main challenge seen in the literature is that PSO tends to stagnate as illustrated in the next section.

III. Illustration of Premature Convergence

A. Premature Convergence and Stagnation

The swarm is said to have *converged prematurely* when the proposed solution approximates a local, rather than global, minimizer and when progress toward a global minimizer has ceased so that continued activity could only hope to refine a local minimizer. *Stagnation* is a result of premature convergence. Once particles have converged prematurely, they continue converging to within extremely close proximity of one another so that the global best and all personal bests are within one miniscule region of the search space. Since particles are continually attracted to the bests in that same small vicinity, particles *stagnate* as the momentum from their previous velocities vanishes. While particles are technically always moving, stagnation can be thought of as a lack of movement discernable on the large scale, from which perspective the stagnated swarm will appear as one static dot.

B. Illustration of Premature Convergence in PSO

The multi-modal Rastrigin function is one of the most difficult benchmarks commonly used in PSO literature because it has many steep wells containing local minima that make the true global minimum difficult to find. PSO can successfully traverse many of the wells containing local minima that would trap a gradient-based method but often gets stuck in high-quality wells near the true global minimizer. In order to illustrate the stagnation problem that has plagued PSO since its original formulation, the search algorithm is applied in this section to minimize the two-dimensional Rastrigin function (Fig. 1: formula in Table III). The parameters used for the two-dimensional demonstration of gbest PSO were: swarm size $s = 10$, acceleration constants $c_1 = c_2 = 1.49$, inertia weight $\omega = 0.72$, and velocity clamping percentage $\lambda = 0.15$. The acceleration coefficients and inertia weight correspond to the values obtained using Clerc's constriction models [10]. The velocity clamping value was selected following [11] and since it has been empirically observed to work well with standard *gbest* PSO [12], at least in the case of thirty dimensional benchmark functions. However, it has also been observed that all PSO parameters seem to be inter-related so that one cannot simply claim $\lambda = 0.15$ to always be best.

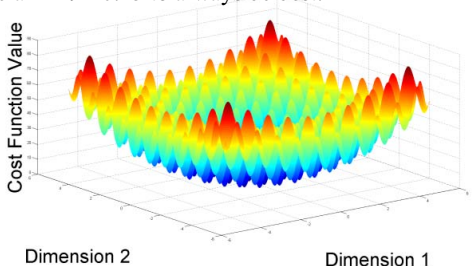

Figure 1. Rastrigin function (Dimensionality $n = 2$).

On this particular two-dimensional example, with a swarm size of ten and the seed of the randomizer set to zero, the trial would converge without premature convergence to the true global minimizer by setting $\lambda = 0.5$. The purpose of this illustration, though, is to provide a visual example of how the swarm can stagnate and be liberated by regrouping. On the thirty-dimensional Rastrigin, PSO prematurely converges for both values.

Swarm motion is graphed on the contour map of the Rastrigin function. In Figs. 2-5, particles can be seen flying from random initialization (Fig. 2) to eventual stagnation at local minimizer $[2,0]$ (Fig. 3-Fig. 5). The true global minimizer of $[0,0]$ is not discovered. A particle finds the relatively quality region near local minimizer $[2,0]$ and communicates this new global best with the rest of the swarm. As the other particles fly in its direction, none finds a better global best, so all converge to $[2,0]$ as their momenta wane.

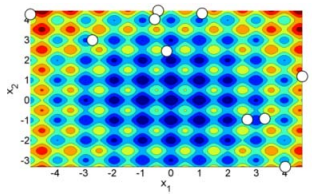

Figure 2. PSO: Particles are randomly initialized within the search space. (Iteration 0)

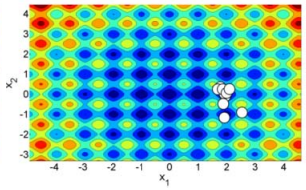

Figure 3. PSO: Particles are converging to local minimizer $[2,0]$ via their attraction to the global best. (Iteration 10)

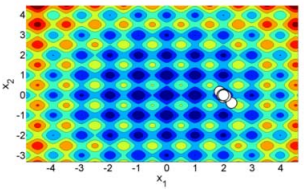

Figure 4. PSO: Premature convergence to a local minimum begins. (Iteration 40)

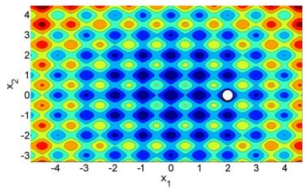

Figure 5. PSO: The local minimum is being honed in on, but no progress is being made toward a better solution since the swarm has stagnated. (Iteration 102)

S*tagnation* is clearly the main obstacle of PSO as little if any progress is made in this state. The goal of the proposed Regrouping PSO (RegPSO) is to detect when particles have prematurely converged and regroup them within a new search space large enough to escape from the local well in which particles have become trapped but small enough to provide an efficient search. It is thought that this will provide an efficient means of escape from local wells so that the swarm can continue making progress rather than restarting. While repeatedly restarting requires running the search an arbitrary number of times, which may or may not be sufficient to approximate true global minimum, RegPSO seeks to improve upon past searches in order to improve the efficiency of approximating the true global minimum.

IV. REGROUPING PSO (REGPSO)

This paper presents an approach to dealing with the stagnation problem by building into the PSO algorithm a mechanism capable of automatically triggering swarm regrouping when premature convergence is detected. The regrouping helps separate particles from the local wells or otherwise sub-optimal solutions to which they converge in order to enable continued progress toward the global minimum. The term *regroup* is defined in Merriam Webster's online dictionary as "*to reorganize (as after a setback) for renewed activity*" [13].

A. Detection of Premature Convergence

In order for swarm regrouping to be triggered, premature convergence has first to be detected. As discussed earlier, all particles are pulled on all dimensions toward the global best via update equations (3)-(6). If none of those particles encounter a better solution (i.e. a new global best) over a period of time, they will continue moving closer to the unchanged global best until they all occupy practically the same location in space and can therefore offer no further improvement. If they actually have happened upon the global minimizer, they may refine that solution by their tiny movements toward it; but in all other cases, it is undesirable for particles to remain in this state. Therefore, it is useful to measure how near particles are to each other so that an effective action can be taken once they have converged to the same region.

Van den Bergh developed and tested various criteria for detecting premature convergence in order to avoid the undesirable state of stagnation [4]: (i) *Maximum Swarm Radius*, which defines stagnation as having occurred when the particle with the greatest Euclidean distance from global best reaches a minimum threshold distance, taken as a percentage of the original *swarm radius*, (ii) *Cluster Analysis*, which terminates the current search when a certain percentage of the swarm has converged to within a pre-specified Euclidean distance of the global best, and (iii) *Objective Function Slope*, which records the number of iterations over which no significant improvement has been seen in the function value returned by the global best and terminates the current search when that number reaches a pre-specified maximum. Van den Bergh found the Maximum Swarm Radius and Cluster Analysis methods to outperform the Objective Function Slope method. In this study, the Maximum Swarm Radius criterion for detecting premature convergence is adopted.

It is proposed herein that when premature convergence is detected using this maximum swarm radius measurement the swarm be regrouped in a new search space centered at the global best as follows.

At each iteration, k, the *swarm radius,* $\delta(k)$, is taken to be the maximum Euclidean distance, in n-dimensional space, of any particle from the global best as follows.

$$\delta(k) = \max_{i \in \{1,\cdots,s\}} \left\| \vec{x}_i(k) - \vec{g}(k) \right\| \qquad (9)$$

where $\|\cdot\|$ denotes the Euclidean norm.

Let $\text{diam}(\Omega) = \|\text{range}(\Omega)\|$ be the diameter of the search space. Particles are considered to be too close to each other and regrouping is triggered when the *normalized swarm radius,* δ_{norm}, satisfies the *premature convergence condition* defined as

$$\delta_{\text{norm}} = \frac{\delta(k)}{\text{diam}(\Omega)} < \varepsilon \tag{10}$$

where ε, called the *stagnation threshold*, is a positive scalar value to be specified by the user. An empirical study shows that selecting $\varepsilon = 1.1\times10^{-4}$ works well with the proposed regrouping mechanism because it allows enough time to pass so that the amount of deviation from global best seen on each dimension can successfully be used as an indicator of the swarm's uncertainty on that dimension while being small enough to allow a decent degree of solution refinement prior to regrouping [12].

B. Swarm Regrouping

When premature convergence is detected as given by condition (10), the swarm is regrouped in a search space centered about the global best that is hoped to be both small enough for efficient search and large enough to allow the swarm to escape from the current well. The regrouping factor

$$\rho = \frac{6}{5\varepsilon}, \tag{11}$$

found to work well across the benchmarks tested, is static across groupings.

Upon detection of premature convergence, the range in which particles are to be regrouped about the global best is calculated per dimension as the maximum of (i) the original range of the search space on dimension j and (ii) the product of the regrouping factor with the maximum distance along dimension j of any particle from global best:

$$\text{range}_j(\Omega^r) = \max\left(\text{range}_j(\Omega^0), \rho \max_{i\in\{1,\cdots,s\}}\left|x_{i,j}^{r-1} - g_j^{r-1}\right|\right). \tag{12}$$

The swarm is then regrouped by re-initializing particles' positions as

$$\begin{aligned} &\vec{x}_i = \vec{g}^{\,r-1} + \vec{r}' \circ \text{range}(\Omega^r) - \frac{1}{2}\text{range}(\Omega^r), \\ &\text{where} \quad \text{range}(\Omega^r) = \left[\text{range}_1(\Omega^r), \cdots, \text{range}_n(\Omega^r)\right] \end{aligned} \tag{13}$$

which utilizes a random vector $\vec{r}'$ to randomize particles within the implicitly defined search space

$$\Omega^r = \left[x_1^{\text{L,r}}, x_1^{\text{U,r}}\right] \times \left[x_2^{\text{L,r}}, x_2^{\text{U,r}}\right] \times \cdots \times \left[x_2^{\text{L,r}}, x_2^{\text{U,r}}\right] \tag{14}$$

with respective lower and upper bounds

$$\begin{aligned} x_j^{\text{L},r} &= g_j^{r-1} - \frac{1}{2}\text{range}_j(\Omega^r), \\ x_j^{\text{U},r} &= g_j^{r-1} + \frac{1}{2}\text{range}_j(\Omega^r). \end{aligned} \tag{15}$$

The swarm regrouping index, r, begins with 0 prior to the occurrence of any regrouping and increments by one with each consecutive regrouping. Vector $\vec{g}^{\,r-1}$ is the global best at the last iteration of the previous grouping, and $\vec{x}_i^{\,r-1}$ is the position of particle i at the last iteration of the prior regrouping. Note that before any regrouping takes place, the original search space, Ω^0, corresponds to a swarm regrouping index of $r = 0$.

The maximum velocity is recalculated with each regrouping according to the new range per dimension as

$$v_j^{\max,r} = \lambda \cdot \text{range}_j(\Omega^r) \tag{16}$$

where λ is the velocity clamping percentage as explained in Section II.

If premature convergence occurs near an edge of the hypercube defining the original search space, the new search space may not necessarily be a subspace of the original search space. Restricting particles to the original search space is easy to do via position clamping or velocity reset [14] if it is known for a fact that better solutions do not lie outside the original search space. In general, it is much easier to make an educated guess as to where a solution will lie than to know for certain that no better solutions can be found elsewhere; for this reason, particles are not generally required to stay within the search space in case they have good reason to explore outside of it. For some applications, however, certain values are known in advance to be impractical to implement.

C. RegPSO Pseudo Code

Do with Each New Grouping
  For j = 1 to n, Calculate $\text{range}_j(\Omega^r)$ and $v_j^{\max,r}$
    For i = 1 to s, initialize velocities
      $v_{i,j} \in \left[-v_j^{\max,r}, v_j^{\max,r}\right]$
    End For
  End For
  For $i = 1$ to s
    Initialize particle positions, $\vec{x}_i$, to lie within Ω^r.
    Initialize personal bests: $\vec{p}_i = \vec{x}_i$.
  End For
  If r = 0
    Initialize global best according to (4)
  End If
  Do Iteratively
    Update velocities according to (6).
    Clamp velocities when necessary.
    Update positions according (5).
    Update personal bests according to (3).
    Update global best according to (4).
    Calculate the swarm radius using (9).
    If (i) the premature convergence criterion of (10) is met or (ii) a user-defined maximum number of function evaluations per grouping is satisfied,
      Then regroup the swarm
    End If
  Until search termination

Figure 6. Pseudo code for the RegPSO algorithm.

Since the PSO algorithm works well initially, the new RegPSO algorithm is not intended to make changes to the original position and velocity update equations, but merely to

liberate the swarm from the well in which it has prematurely converged using an automatic regrouping mechanism. The pseudo code of RegPSO is given in Fig. 6.

D. Illustration of the Regrouping Mechanism

RegPSO was applied to the two-dimensional Rastrigin function to illustrate its regrouping mechanism. Stagnation was detected by RegPSO in Fig. 5, prompting the swarm to regroup as shown in Fig. 7. Figs. 7 & 11 illustrate the efficient regrouping mechanism by which the swarm can escape from local minima and continue its search without needing to restart on the entire search space. Fig. 9 shows that the swarm seemed less certain about its search along the horizontal dimension: Fig. 11 shows that the swarm consequently regrouped within a larger range on the horizontal dimension than on the vertical – accounting for this uncertainty in a computationally simple manner so as not to overly complicate an algorithm that has computational simplicity as its main strength. This is the idea behind using the maximum deviation from global best on each dimension to determine the new range per dimension. Fig. 12 shows that the swarm has converged to the global minimum. Fig. 13 compares the performance of standard gbest PSO and RegPSO on the two-dimensional Rastrigin benchmark.

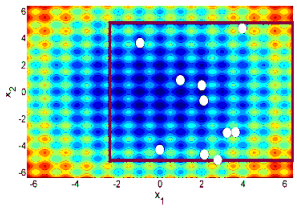

Figure 7. RegPSO: Regrouping PSO detects premature convergence at iteration 102 of Fig. 5 once gbest PSO has stagnated. The swarm is regrouped and continues making progress toward the global minimum. (Iteration 103)

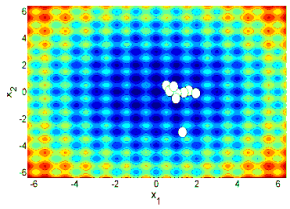

Figure 8. RegPSO: The swarm is migrating toward a better position found by one of the particles near position [1, 0]. (Iteration 123).

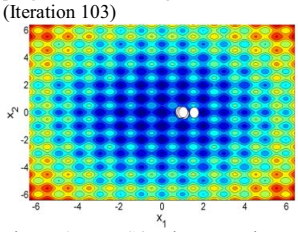

Figure 9. RegPSO: The swarm is prematurely converging to a new local minimum. (Iteration 143)

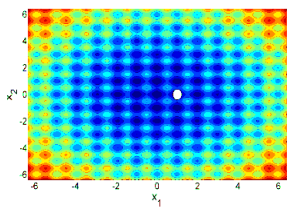

Figure 10. RegPSO: The swarm has stagnated at local minimizer $[1, 0]$ (Iteration 219)

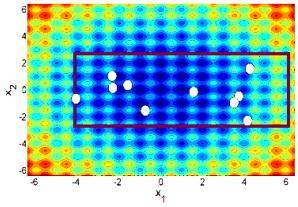

Figure 11. RegPSO: A second regrouping is triggered. (Iteration 220)

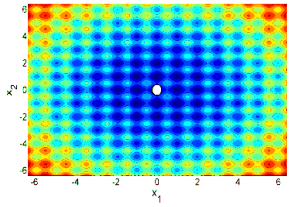

Figure 12. RegPSO: The global minimum has been found successfully. (Iteration 270)

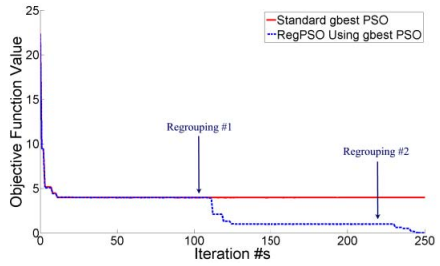

Figure 13. Comparison of the performances of standard gbest PSO and the proposed RegPSO on the Rastrigin benchmark function with dimensionality $n = 2$.

In RegPSO, particles are regrouped efficiently when the swarm prematurely converges so that the search can continue rather than restarting.

V. Simulation Results

Firstly, RegPSO is compared to the standard gbest and lbest PSO algorithms and OPSO in order to demonstrate by comparison how effectively it avoids stagnation. Secondly, RegPSO is compared with MPSO in order to demonstrate its improved efficiency over continually restarting – even though an improved local minimizer in GCPSO is used with each restart of MPSO. Tables I and II use a different number of function evaluations to give the reader an idea of how RegPSO improves solution quality over time.

In Table I, 800,000 function evaluations are used to compare gbest PSO, lbest PSO, OPSO and RegPSO. The point of selecting this number is to show that RegPSO is capable of solving the stagnation problem and continuing onward to approximate the true global minimizer if given enough time to do so. The results on Rastrigin are especially impressive since this benchmark generally returns high function values in the literature due to stagnation of the swarm. The mean performance across benchmarks is displayed in the final row, from which it can be seen that RegPSO is more consistent across benchmarks. RegPSO provides significantly better consistency across the multi-modal benchmarks than the comparison algorithms while still performing well on the simpler uni-modal functions that do not require regrouping. That RegPSO is able to approximate the true global minimum on all 400 trials across benchmarks evidences that it is a good general purpose optimizer.

In Table II, 200,000 function evaluations are used to compare with MPSO on the multi-modal functions using the normalized swarm radius convergence detection criterion. MPSO was selected for being Van den Bergh's best-performing restart algorithm outperforming guaranteed convergence PSO (GCPSO), multi-start PSO using the cluster analysis convergence detection technique ($MPSO_{cluster}$), multi-start PSO using the objective function slope convergence detection technique ($MPSO_{slope}$), and random particle swarm optimization (RPSO) [4]. It is worth noting that after only 200,000 function evaluations in Table II, RegPSO has already minimized Rastrigin better than the comparison algorithms of Table I were able to do after 800,000 function evaluations.

In both tables, the mean performance of RegPSO across benchmarks was superior to that of the comparison algorithms.

VI. Conclusions

This paper has presented an approach for overcoming the stagnation problem of PSO by building into the algorithm a mechanism that can automatically trigger swarm regrouping when premature convergence is detected. This regrouping helps liberate the swarm from the state of premature convergence in order to enable continued progress toward true global minimum. RegPSO has been shown to have better mean performance than the algorithms compared with – a result that would have been more pronounced had only multi-modal test functions been used. RegPSO also consistently outperformed in the presence of noise. Given sufficient function evaluations, RegPSO was able to solve the stagnation problem for each benchmark utilized.

TABLE I. Comparison of lbest PSO, gbest PSO, OPSO and RegPSO.

Statistics results from 50 trials per benchmark per algorithm over 800,000 function evaluations per trial using swarm size 20, $\lambda = 0.5, c_1 = c_2 = 1.49618,$ and $\omega = 0.72984.$ RegPSO was used with $\varepsilon = 1.1\times10^{-4}$ and $\rho = 1.2\varepsilon^{-1}$ and each grouping was limited to a maximum of 100,000 function evaluations before regrouping. Since the same number of trials were used per benchmark per algorithm, the mean of each algorithm's means per benchmark produces the overall mean performance of the algorithm across benchmarks. **Bold values** are the best in their rows.

Benchmark		gbest PSO	lbest PSO (neighborhood of size 2)	OPSO	RegPSO
Ackley	Median:	3.7069	**7.9936E-15**	2.7385	4.4632E-7
	Mean:	3.9115	0.075469	2.6724	**4.6915E-7**
	Min:	0.9313	**7.9936E-15**	4.4409E-15	1.606E-7
	Max:	8.6427	1.5017	5.1581	**8.7023E-7**
	Std. Dev.:	1.7067	0.30754	0.98749	**1.4519E-7**
Griewangk	Median:	0.049122	**0.007396**	0.012321	0.0098573
	Mean:	0.055008	**0.0093997**	0.025749	0.013861
	Min:	0	0	0	0
	Max:	0.15666	**0.054069**	0.078513	0.058867
	Std. Dev.:	0.044639	**0.011964**	0.026621	0.01552
Quadric	Median:	**1.6824E-79**	6.7942E-13	9.1363E-70	2.5503E-10
	Mean:	**4.1822E-75**	1.0389E-11	8.2046E-65	3.1351E-10
	Min:	**4.146E-84**	8.2216E-16	1.8778E-72	6.0537E-11
	Max:	**2.0732E-73**	1.6906E-10	3.7704E-63	9.5804E-10
	Std. Dev.:	**2.9314E-74**	3.0378E-11	5.3377E-64	2.2243E-10
Quartic with Noise	Median:	0.00272	0.012554	0.0009416	**0.0006079**
	Mean:	0.0039438	0.01325	0.0010166	**0.00064366**
	Min:	0.00060861	0.0048734	0.00039321	**0.0002655**
	Max:	0.019695	0.029155	0.0020241	**0.0012383**
	Std. Dev.:	0.0040209	0.0048053	0.00035806	**0.00021333**
Rastrigin	Median:	70.64194	54.2252	65.66717	**2.3981E-14**
	Mean:	71.63686	54.2849	66.16463	**2.6824E-11**
	Min:	42.78316	25.8689	30.84371	**0**
	Max:	116.4097	85.5663	109.4452	**1.3337E-9**
	Std. Dev.:	17.1532	15.5614	17.23225	**1.886E-10**
Rosenbrock	Median:	5.35546E-9	3.90016	**4.4423E-10**	0.0030726
	Mean:	2.06915	3.25523	1.8641	**0.0039351**
	Min:	**2.68986E-18**	4.1282E-5	2.13491E-15	1.7028E-5
	Max:	13.315	19.0917	18.411	**0.018039**
	Std. Dev.:	3.1387	3.11484	2.83496	**0.0041375**
Spherical	Median:	**0**	1.4606E-169	**0**	5.8252E-15
	Mean:	2.4703E-323	5.5139E-160	**9.8813E-324**	9.2696E-15
	Min:	**0**	1.3621E-177	**0**	1.2852E-15
	Max:	8.745E-322	2.7569E-158	2.4703E-322	4.9611E-14
	Std. Dev.:	**0**	3.8988E-159	**0**	8.6636E-15
Weighted Sphere	Median:	**0**	4.0093E-168	**0**	8.1295E-14
	Mean:	1.0869E-321	5.6269E-162	**9.8813E-324**	9.8177E-14
	Min:	**0**	1.1076E-176	**0**	1.9112E-14
	Max:	5.3903E-320	1.6576E-160	**3.2608E-322**	2.5244E-13
	Std. Dev.:	**0**	2.6112E-161	**0**	5.4364E-14
Mean Performance	Mean of Means:	9.7096	7.2048	8.8410	**2.305E-3**

TABLE II. COMPARISON OF MPSO AND REGPSO

Statistics result from 50 trials per benchmark per algorithm over 200,000 function evaluations per trial using a swarm size of 20, $\lambda = 0.5, c_1 = c_2 = 1.49,$ and $\omega = 0.72.$ RegPSO was used with $\varepsilon = 1.1\times10^{-4}$ and $\rho = 1.2\varepsilon^{-1}$ and each grouping was limited to a maximum of 100,000 function evaluations before regrouping. **Bold values** are the best in their rows. Comparison data can be found in [4].

Benchmark		MPSO (using GCPSO)	RegPSO (using standard gbest PSO)
Ackley	Median:	0.931	**4.6643E-6**
	Mean:	0.751	**5.1857E-6**
Griewangk	Median:	**1.52E-9**	0.019684
	Mean:	**1.99E-9**	0.028409
Rastrigin	Median:	45.8	**3.9798**
	Mean:	45.8	**4.3208**
Mean Performance	Mean of Means:	15.517	**1.4497**

TABLE III. BENCHMARK FUNCTIONS TESTED

Benchmark		Dimensionality n
Ackley	$f(\vec{x}) = 20 + e - 20e^{-0.2\sqrt{\frac{\sum_{j=1}^{n} x_j^2}{n}}} - e^{\frac{\sum_{j=1}^{n}\cos(2\pi x_j)}{n}}$ $-30 \le x_j \le 30$	30
Griewangk	$f(\vec{x}) = 1 + \sum_{j=1}^{n} \frac{x_j^2}{4000} - \prod_{j=1}^{n} \cos\left(\frac{x_j}{\sqrt{j}}\right)$ $-600 \le x_j \le 600$	30
Quadric	$f(\vec{x}) = \sum_{j=1}^{n} \left(\sum_{k=1}^{j} x_j \right)^2$ $-100 \le x_j \le 100$	30
Quartic with Noise	$f(\vec{x}) = random[0,1) + \sum_{i=1}^{n} i \cdot x_i^4$ $-1.28 \le x_j \le 1.28$	30
Rastrigin	$f(\vec{x}) = 10n + \sum_{j=1}^{n} (x_j^2 - 10\cos(2\pi x_j))$ $-5.12 \le x_j \le 5.12$	30
Rosenbrock	$f(\vec{x}) = \sum_{j=1}^{n-1} \left(100(x_{j+1} - x_j^2)^2 + (1 - x_j)^2\right)$ $-30 \le x_j \le 30$	30
Spherical	$f(\vec{x}) = \sum_{j=1}^{n} x_j^2$ $-100 \le x_j \le 100$	30
Weighted Sphere	$f(\vec{x}) = \sum_{j=1}^{n} j \cdot x_j^2$ $-5.12 \le x_j \le 5.12$	30

REFERENCES

[1] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, Piscataway, NJ., 1995, pp. 1942-1948.

[2] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan, 1995, pp. 39-43.

[3] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*: John Wiley & Sons, 2006.

[4] F. Van den Bergh, "An analysis of particle swarm optimizers," Ph.D. Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.

[5] F. Van den Bergh and A. P. Engelbrecht, "A new locally convergent particle swarm optimiser," in *Proceedings of the IEEE Conference on Systems, Man, and Cybernetics*, Hammamet, Tunisia, 2002, pp. 96-101.

[6] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 1999, pp. 1931-1938.

[7] R. C. Eberhart*, et al.*, *Computational Intelligence PC Tools*: Academic Press Professional, 1996.

[8] H. Wang*, et al.*, "Opposition-based particle swarm algorithm with Cauchy mutation," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2007, pp. 4750-4756.

[9] R. Marti, "Multi-start methods," in *Handbook of Metaheuristics*, F. W. Glover and G. A. Kochenberger, Eds., Springer, 2003, pp. 355-368.

[10] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in multidimensional complex space," *IEEE Transactions on Evolutionary Computation,* vol. 6, pp. 58-73, 2002.

[11] B. Liu*, et al.*, "An improved particle swarm optimization combined with chaos," *Chaos, Solitons and Fractals,* vol. 25, pp. 1261-1271, 2005.

[12] G. Evers, "An automatic regrouping mechanism to deal with stagnation in particle swarm optimization," M.S. Thesis, Electrical Engineering Department, The University of Texas-Pan American, Edinburg, TX, 2009.

[13] *Merriam Webster's Online Dictionary*, 2009.

[14] G. Venter and J. Sobieski, "Particle swarm optimization," in *Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Stuctures, Structual Dynamics, and Materials Conference*, Denver, CO, 2002, pp. 1-9.